주메모리 DBMS를 위한 다계층 자바 API 설계 및 구현

이도영○ 김영국 진성일
충남대학교 컴퓨터과학과
{dylee, ykim, sijin}@cs.cnu.ac.kr

Design and Implementation of Multi-level Java API for Main Memory DBMS

Do-Young Lee○ Young-Kuk Kim Seong-Il Jin
Dept. of Computer Science, Chungnam National University

요 약

데이터베이스 시스템은 다양한 표준 인터페이스들을 제공함으로써 개발자에게 편이성과 시스템의 변화에 독립적인 접근방식을 제공하여 시스템 구축의 생산성을 향상시켜왔다. 특히 JDBC를 제공하는 자바언어는 플랫폼에 독립적인 실행 환경을 지원하고 인터넷 환경에 적합하게 설계되어 있기 때문에 많은 데이터베이스 응용 시스템이 자바언어로 구축되고 있다. 한편, 인터넷 환경에서의 고성능을 요하는 응용 분야가 확대됨에 따라 빠른 데이터 접근시간을 가진 주메모리 DBMS 중요성이 부각되고 있다. 본 논문에서는 주메모리 DBMS가 상이한 성능 요구조건을 갖는 다양한 실시간 응용분야에 융통성 있게 적용될 수 있도록 이를 위한 다계층 자바 API를 설계 및 구현하였다.

1. 서론

데이터베이스 시스템은 자료를 구조적으로 저장, 관리, 조작할 수 있는 시스템을 말한다. 데이터베이스 시스템은 여러 응용분야에서 자료를 효과적으로 사용할 수 있게 함으로써 필수 구성요소로 자리잡았다. 그러한 추세에 따라서 다양한 플랫폼에서 수행될 수 있는 여러 데이터베이스 시스템이 개발되었고 인터넷이 보편화됨에 따라서 제공되고 있는 자료의 양이 방대해지고 다수 클라이언트들을 수용할 수 있도록 고성능을 요구하는 인터넷 응용 시스템이 증가함에 따라 평균 탐색시간이 빠른 주기억장치를 활용한 DBMS가 각광을 받고 있다. 또한 데이터베이스 시스템 제공자들은 응용 시스템 개발자들에게 쉽고, 시스템의 변화에 독립적인 접근을 할 수 있도록 ODBC나 JDBC와 같은 표준화된 인터페이스를 제공하게 되었다. 이러한 표준화된 인터페이스의 제공은 데이터베이스 시스템의 활용성을 배가시키고 응용 시스템 구축의 생산성을 극대화시킬 수 있다. 특히 DBMS를 위한 자바 API인 JDBC는 Sun에서 제공되고 있는 데이터베이스 시스템을 위한 표준 인터페이스로서, 플랫폼에 독립적이기 때문에 시스템간의 호환성을 제공하며, 인터넷 환경에 적합하게 설계되어 있는 자바를 사용하여 많은 응용 시스템들이 구축되고 있다. 그러나, 이러한 장점에도 불구하고 자바는 느린 성능으로 인하여 빠른 처리 속도를 요구하는 실시간 응용에는 잘 쓰이지 않고 있다.

본 논문에서는 플랫폼 독립적인 자바의 장점을 살리면서도 상이한 성능 요구조건을 갖는 다양한 실시간 응용 프로그램의 개발을 지원할 수 있도록 주메모리 DBMS를 위한 다계층(multi-level) 자바 인터페이스(JDBC)를 개발하게 되었다. 주메모리 DBMS가 사용되는 응용 분야에서는 자료의 고속 처리를 요구하기 때문에 순수 자바 언어로 구현된 JDBC 인터페이스는 높은 호환성을 제공할 수 있지만 응용 시스템의 성능을 저하시킬 수 있는 요인이 된다. 따라서, 고속 처리를 요구하는 응용 분야를 위해서는 JNI(Java Native Interface)를 활용한 저계층(low-level) API와 높은 호환성을 위해서는 순수 자바로 구현된 고계층(high-level) API를 동시에 제공하는 다계층 자바 API를 설계 및 구현하였다.

본 논문의 구성은 다음과 같다. 2절에서 기존의 범용 데이터베이스 시스템을 위한 표준 인터페이스에 대해서 살펴보고, 본 논문의 개발 대상 주메모리 DBMS인 Kairos에 대해서 기술한다. 3절에서는 Kairos에서 JDBC 인터페이스를 제공하기 위해 Kairos SQL 인터페이스를 살펴보고, 주메모리 DBMS를 위한 JDBC 인터페이스 설계 및 구현 내용에 대해 기술하고, 마지막으로 결론 및 향후 과제에 대해서 살펴본다.

2. 관련 연구

이 절에서는 기존의 범용 데이터베이스 시스템을 위한 대표적인 표준 인터페이스에 대해 살펴보고, 기반 시스템인 Kairos에 대해서 간략하게 기술한다.

2.1 표준 인터페이스

데이터베이스 시스템을 위한 표준 인터페이스의 역사는 SQL에서 시작되었다. 하지만 SQL은 실제 프로그램을 작성하기 위한 언어가 아니었고, 이를 프로그램 작성에 적용할 수 있는 Embedded SQL이나 X/Open CLI(Call Level Interface)로 발전하게 되었다. 이를 기반으로 현재의 ODBC나 JDBC와 같은 표준 인터페이스들이 등장하게 된 것이다.

- ODBC

ODBC(Open DataBase Connectivity)는 마이크로소프트사에서 제공하는 표준 인터페이스이다. 윈도우 플랫폼에 한정되기는 하지만, 91년도에 발표한 ODBC 표준 인터페이스는 향상된 오류 처리와 실행 일괄 처리, 조작 가능한 커서 작업을 제공하는 ODBC 3.5 에 이르고 있다. 이 ODBC를 사용하는 응용 프로그램은 ODBC 드라이버 관리자를 통해 ODBC 드라이버를 제공하는 데이터베이스 시스템에 접근 할 수 있다. 드라이버 관리자는 접근하고자 하는 데이터베이스의 ODBC 드라이버를 설치, 적재하고 사용할 수 있도록 한다. 이러한 ODBC는 데이터베이스와 응용 시스템간의 자료형에 대한 투명성을 제공해 주며, 실행 오류에 대한 일관적인 표현을 가능하게 해준다.[1]

ODBC 드라이버의 형태는 구현 계층 모델에 따라 크게 3가지로 나눌 수 있다. 1-Tier Driver는 다중 연결과 트랜잭션을 지원하지 않는 간단한 형태의 드라이버로써 드라이버 자체가 파일과 연동되어 데이터베이스의 엔진 역할을 수행할 수 있는 드라이버이다. 2-Tier Driver는 클라이언트/서버 구조의 드라이버로써 네트워킹 프로토콜을 사용하여 데이터베이스 시스템의 네이티브 API나 데이터베이스 시스템에서 제공되는 ODBC 인터페이스를 통해 연결 가능한 드라이버 형태이다. 3-Tier Driver는 클라이언트/서버 구조 사이에 게이트웨이가 존재하여 여러 데이터베이스 시스템과의 연결을 중재해주는 형태의 드라이버이다.

- JDBC

JDBC는 Sun에서 제공하고 있는 데이터베이스 표준 인터페이스이다. 기본적인 기능만을 제공하는 JDBC 1.0에서 커서 조작기능과 일괄 처리 작업 지원, 다중 연결의 과부하를 줄이기 위한 연결 풀 기능, 새로운 자료형 추가 등을 제공하는 JDBC 2.1까지 출시되어 있고 향상된 연결 풀 설정 기능과 질의 전송에 사용되는 객체인 Statement 풀링 등 고급화된 기능이 추가된 JDBC 3.0이 출시될 예정이다[2].

JDBC 드라이버는 구현하는 방법에 따라 크게 네 가지 타입으로 구분되는데, 타입 1은 ODBC 드라이버를 통해 데이터베이스와 자바 응용 프로그램을 연결하는 JDBC-ODBC Bridge Driver이고, 타입 2는 하부 구현이 해당 데이터베이스에 의존적인 API를 사용한 Native-API partly-Java Driver이며, 타입 3은 자바만으로 구현된 드라이버가 미들웨어를 통해서 데이터베이스와 연결하는 JDBC-Net pure Java Driver이고 타입 4는 데이터베이스와 네트워크 프로토콜을 사용하여 직접 연결하는 Native-protocol pure Java Driver로 구분할 수 있다[3]. 타입 1과 타입 2는 플랫폼에 따른 호환성을 제공할 수는 없지만 타입 2는 본래의 프로토콜을 그대로 사용하기 때문에 빠른 성능을 제공할 수 있다. 타입 3과 타입 4는 처리속도가 떨어지는 반면 자바만으로 구현되어 높은 호환성을 제공한다. 본 논문에서 구현하는 JDBC 드라이버 타입은 타입 2와 타입 4를 제공한다.

2.3 Kairos

Kairos는 현재 (주)리얼타임테크에서 개발한 주메모리 DBMS이다. 주메모리 DBMS란 주기억장치에 모든 데이터를 영구히 저장하는 시스템으로써 컴퓨터 하드웨어의 발전으로 인한 대용량 메모리 추세와 가격의 하락으로 주기억 상주 데이터베이스의 개발이 현실화되었으며 항공 제어 시스템이나, 네트웍 관련 시스템과 같은 빠른 데이터 처리를 요구하는 응용 분야에 적합한 데이터베이스이다.

Kairos는 기억장치 관리, 트랜잭션 관리, 동시성 제어, 회복 등의 기능을 제공하며, 실시간 응용 시스템에 따라 각 기능을 선택할 수 있는 융통성 있는 데이터베이스 구조를 가진다. Kairos 동작 구조는 다음과 같다. 서버 프로세스는 동시에 여러 개의 트랜잭션을 처리할 수 있는 다중 쓰레드 구조로 동작하며, 사용자 프로세스와의 통신을 위해 메시지 큐를 사용한다. 서버 프로세스의 쓰레드는 트랜잭션을 분배하는 트랜잭션 디스패처 쓰레드(Transaction Dispatcher Thread)와 실제 트랜잭션을 수행하는 트랜잭션 처리 쓰레드(Transaction Processing Thread) 등으로 구성되며, 트랜잭션 디스패처 쓰레드와 트랜잭션 처리 쓰레드간의 통신은 트랜잭션 큐를 통하여 이루어진다.

3. 주메모리 DBMS를 위한 JDBC 인터페이스 설계 및 구현

Kairos 시스템은 현재 표준 SQL-92의 주요기능을 지원하고 있다. Kairos SQL은 클라이언트 라이브러리인 CLI(Call Level Interface)와 Kairos 서버의 질의 처리 관리자에 의해 수행된다. CLI는 질의를 질의 처리 관리자에 전달하고 질의 수행 결과의 유지 및 상위 인터페이스에 이를 전달하는 역할을 한다. 질의 처리 관리자는 CLI로부터 전달된 질의를 수행하여 그 결과를 CLI에게 전달하게 된다. 자바 응용 프로그램을 지원하기 위한 Kairos의 JDBC 인터페이스는 질의 처리관리자와의 연결과 질

의 전달, 질의 결과 저장을 수행 할 수 있도록 JNI를 사용한 타입 2의 형태와 자바만으로 구현된 타입 4의 형태로 구현한다.

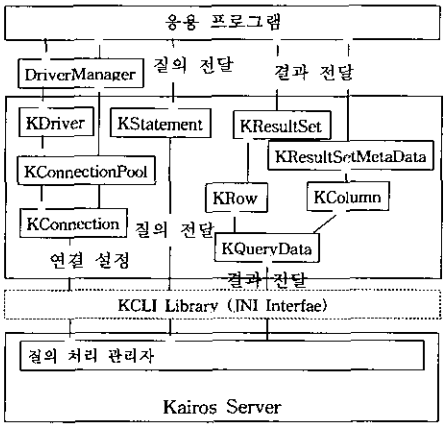
본 연구에서 구현한 JDBC 인터페이스는 JDBC 2.0을 기반으로 하고 있으며 구현된 Kairos JDBC 인터페이스는 [표 1] 과 같다.

[표 1] Kairos JDBC 인터페이스

JDBC	Kairos
Driver	KDriver
Connection	KConnection
Statement	KStatement
PreparedStatement	KPreparedStatement
CallableStatement	-
ResultSet	KResultSet
ResultSetMetaData	KResultSetMetaData
DatabaseMetaData	KDatabaseMetaData
JDBC 2.0 DataType	-

각 클래스들의 역할을 살펴보면, KDriver는 드라이버 관리자에 의해 로딩되며 응용 프로그램과 질의 처리 관리자와의 연결을 준비하는 역할을 하며 KConnection은 질의 처리 관리자와의 연결을 담당하고 KStatement와 KPreparedStatement는 질의 처리 관리자에게 질의를 전달하고 결과를 수신한다. KResultSet은 질의 수행 결과인 결과 셋을 저장하며 KResultSetMetaData는 결과 셋에 대한 메타 정보를 유지하고 KDatabaseMetaData는 데이터베이스의 메타 정보를 저장하는 역할을 한다.

Kairos에서의 다계층 자바 API가 질의 처리 관리자와 동작하는 구조는 [그림 1]과 같다.



[그림 1] Kairos 다계층 자바 API 의 동작 과정

위의 구조는 타입 2와 타입 4의 형태를 모두 포함하고 있다. 고성능을 요구하는 응용 시스템에 적용하고자 하는 경우, 연결을 위한 KConnection과 질의 전송을 위한 KStatement, 그리고 질의 수신을 위한 KQueryData 클래스는 JNI를 사용하여 구현된 KCLI Library를 통해서 질의 처리 관리자와 질의 처리를 할 수 있도록 구현하였다. 또한 고성능보다는 호환성을 위한 응용 시스템인 경우 KCLI Library를 활용하지 않고 모든 클래스가 자바로 구현되어 다른 플랫폼에서도 실행할 수 있는 호환성을 제공할 수 있도록 하였다.

동작 과정을 살펴 보면, 자바 응용 프로그램에서 드라이버 관리자를 통해 KDriver를 로딩하여 KConnection 객체를 생성한다. KConnection은 질의 처리 관리자와 연결을 설정한다. KConnectionPool은 기존 연결 객체를 다른 클라이언트가 재사용 가능하게 함으로써 연결 설정의 소요 시간을 단축시키고 다수의 연결과 해제로 인한 성능 오버헤드를 줄이기 위해서 Connection Pooling 기법을 제공한다[4].

응용 프로그램은 KStatement를 통해 질의를 질의 처리 관리자에게 전송하고, KStatement는 질의에 대한 결과 셋이 존재하는 경우, KQueryData를 통해 결과 셋을 수신하도록 하고 KRow들로 구성된 KResultSet과 KColumn들로 구성된 KResultSetMetaData로 재구성하여 저장한다.

4. 결론 및 향후 방향

본 논문에서는 주메모리 DBMS인 Kairos가 자바 응용 프로그램을 지원할 수 있도록 다계층 자바 API를 설계 및 구현하여 고성능을 요구하는 시스템과 호환성을 필요로하는 시스템을 모두 지원할 수 있게 되었다. 향후, 커서 기능 및 지원 데이터 타입 등, 미흡한 부분들을 보완하고, JDBC 2.0의 확장 패키지를 지원할 것이다.

5. 참고 문헌

[1] ODBC Document - http://www.microsoft.com/data/odbc/prodinfo.htm
[2] JDBC 3.0 - http://java.sun.com/products/jdbc/
[3] JDBC Guide - http://java.sun.com/j2se/1.3/docs/guide/jdbc/
[4] Connection Pooling - http://developer.java.sun.com/developer/onlineTraining/Programming/JDCBook/conpool.html